Queen Mary
University of London

MAS 335 Cryptography

Notes 4: Stream ciphers, continued Spring 2008

Recall from the last part the definition of a stream cipher:

Definition: A *stream cipher* over an alphabet of q symbols $a_1, \ldots, a_q$ requires a *key*, a random or pseudo-random string of symbols from the alphabet with the same length as the plaintext, and a *substitution table*, a Latin square of order q (whose entries are symbols from the alphabet, and whose rows and columns are indexed by these symbols). If the plaintext is $p = p_1 p_2 \ldots p_n$ and the key is $k = k_1 k_2 \ldots k_n$, then the ciphertext is $z = z_1 z_2 \ldots z_n$, where $z_t = p_t \oplus k_t$ for $t = 1, \ldots, n$; the operation $\oplus$ is defined as follows:

> $a_i \oplus a_j = a_k$ if and only if the symbol in the row labelled a_i and the column labelled a_j of the substitution table is a_k.

We extend the definition of $\oplus$ to denote this coordinate-wise operation on strings: thus, we write $z = p \oplus k$, where p, k, z are the plaintext, key, and ciphertext strings.

We also define the operation $\ominus$ by the rule that $p = z \ominus k$ if $z = p \oplus k$; thus, $\ominus$ describes the operation of decryption.

Fish

A simple improvement of the Vigenère cipher is to encipher twice using two different keys k_1 and k_2. Because of the additive nature of the cipher, this is the same as enciphering with $k_1 + k_2$. The advantage is that the length of the new key is the least common multiple of the lengths of k_1 and k_2. For example, if we encrypt a message once with the key `FOXES` and again with the key `WOLVES`, the new key is obtained by encrypting a six-fold repeat of `FOXES` with a five-fold repeat of `WOLVES`, namely

`BCIZWXKLPNJGTSDASPAGQJBWOTZSIK`

The new key has period 30. Re-encrypting with a word of length 7 would have the effect that the new key has period 210.

This idea was exploited in the Second World War German cipher codenamed "Fish", so-called because it used the Siemens T52 machine known as *Sägefisch* (sawfish). This cipher, which was broken by the Bletchley Park cryptanalysts, is less well-known than the Enigma cipher, but is probably of even greater significance, since it was used for strategic messages, troop dispositions, etc., between the German High Command and the theatres of war. The book *Code Breakers: The Inside Story of Bletchley Park* (edited by F. H. Hinsley and Alan Stripp) gives more detail about breaking this cipher, which has been described as the greatest intellectual achievement of the war.

The Fish cipher employed the 5-bit International Telegraph Code, described in Part 3 of the notes. The five bits of each character in the plaintext were separated into five bitstreams which were enciphered separately and then reassembled into a sequence of 5-bit words for transmission.

The encryption of each substream was by means of a stream cipher, generated by a mechanical device. The first stage consisted of one Vigenère cipher for each substream; the periods of these ciphers were 41, 31, 29, 26 and 23. Each cipher was implemented by a toothed wheel; the teeth could be extended or retracted, corresponding to a 1 or a 0 in the corresponding keyword. The wheels advanced one place after encrypting one bit from each stream in parallel. This was followed by a second cipher, like a Vigenère cipher but where we sometimes advance to the next letter of the keyword and sometimes remain with the same one, depending on the operation of two further wheels. The periods of the second ciphers were 43, 47, 51, 53 and 59, while the control wheels had periods 37 and 61. (The precise method of operation, and a diagram of the machine, appear in the book *Code Breakers*.)

Since the wheel sizes are pairwise coprime, the period of the keystring generated by such a cipher is their product:

$$23 \cdots 61 = 16\,033\,955\,073\,056\,318\,658.$$

The keys of the different Vigenère ciphers and the control wheels could be set, but the lengths of the wheels was fixed.

It would have been possible for the Bletchley Park cryptanalysts to have assembled models of the cipher machines. But they felt that the supply of parts for such machines would have drawn attention to the fact that they were attempting to break the cipher. So instead they built electronic machines (including Colossus, the first stored-program computer) out of readily available parts used for telephone switchgear. This move from mechanical to electronic methods in cryptography was probably the most significant result of the Bletchley Park codebreakers.

Shift registers

Ideally the key string for a stream cipher should be as unpredictable as possible; but for practical purposes we need a simple way to generate it. One method which has been widely used for generating "pseudo-random" binary sequences involves shift registers.

Figure 1 shows a shift register.

Each of the boxes in the shift register contains one bit (zero or one). The shift register is controlled by a clock which ticks at discrete time intervals. When the clock ticks, the contents x_0 and x_1 of the first two boxes are added (mod 2); then the contents of each box is shifted one place left (that of the first box is output) and the result of the addition is put in the last box.

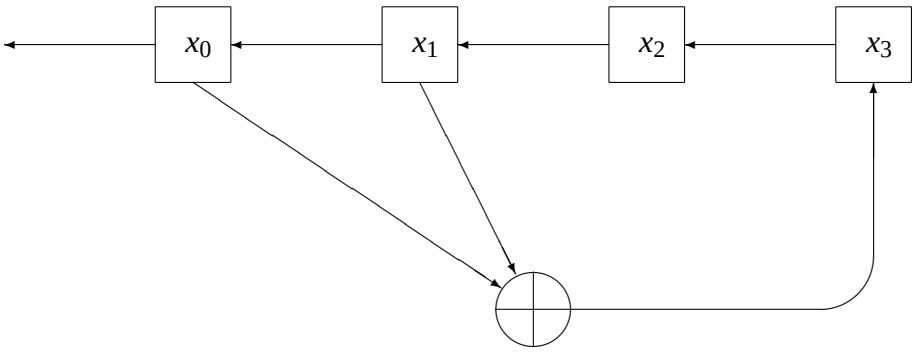

Figure 1: A shift register

Suppose that the boxes initially contain 0001. Then, at successive clock ticks, they become 0010, 0100, 1001, 0011, 0110, 1101, 1010, 0101, 1011, 0111, 1111, 1110, 1100, 1000, 0001, and the machine outputs the sequence

000100110101111

At this point, the contents have returned to their original values, and the machine then repeats the same cycle indefinitely.

We see that, for this particular shift register, every possible binary 4-tuple except 0000 occurs precisely once in a cycle as the contents of the boxes. Moreover, the contents of the boxes at stage n become the next four bits of the output string. So, if we consider the string as continuing indefinitely, and if we look at it through a window

which shows just four bits at a time, then we see each of the $2^4 - 1 = 15$ non-zero 4-tuples just once in each cycle. Note that we could start with any non-zero 4-tuple and the same cycle would be obtained.

On the other hand, if we start with 0 in each box, then the contents of the boxes will always be 0, and the output string consists entirely of zeros – not very good as a pseudo-random string.

In general, a shift register works in the same way. It is specified by giving

(a) the number of boxes;

(b) which boxes are connected to the "adder".

If there are n boxes, we speak of an n-bit shift register. Its configuration at any given time is the binary n-tuple giving the contents of the boxes at that time.

For reasons that will become clear in the next section, it is convenient to describe a shift register by a polynomial over the binary field. The degree n of the polynomial is the number of boxes; the coefficient of x^i is 1 if $i = n$ or if the ith box is connected to the adder, and 0 otherwise. (We number the boxes from 0 on the left to $n-1$ on the right.) Thus, the polynomial describing the shift register in Figure 1 is

$$x^4 + x + 1.$$

Proposition 5 *Suppose that a shift register is described by the polynomial*

$$x^n + \sum_{i=0}^{n-1} a_i x^i.$$

Then its output sequence is given by the recurrence relation

$$x_{k+n} = \sum_{i=0}^{n-1} a_i x_{k+i}.$$

Proof: Suppose that the configuration is $(u_0, \ldots, u_{n-1})$. At the next clock tick, the adder computes $t = \sum_{i=0}^{n-1} a_i u_i$. The next n bits output are, in order, $x_k = u_0$, $x_{k+1} = u_1$, $\ldots$, $x_{k+n-1} = u_{n-1}$, $x_{k+n} = t$. Hence the sequence is given by the recurrence relation.

An n-bit shift register (one with n boxes $x_0, \ldots, x_{n-1}$) which starts in a non-zero configuration must return to its starting point in at most $2^n - 1$ steps, since there are exactly this many non-zero configurations it can have. Thus, its period is at most $2^n - 1$. An n-bit shift register is said to be *primitive* if is period is $2^n - 1$; that is, if it has the property that, if the starting configuration is non-zero, then each of the $2^n - 1$ non-zero n-tuples occurs once as a configuration in the course of a cycle. The next theorem asserts that primitive shift registers exist with any given number of bits.

Theorem 6 *For any positive integer n, there is a primitive n-bit shift register.*

Of course, it is easy to construct a shift register with a moderate number of bits, say 30 or 100. We can ensure (by choosing a primitive shift register) that its output sequence will not repeat during the lifetime of the universe!

Algebraic formulation

The behaviour of the shift register can be described algebraically. If $x = (x_0, x_1, x_2, x_3)$ are the contents of the shift register at any moment, and $y = (y_0, y_1, y_2, y_3)$ the contents after the clock ticks, then we have

$$\begin{array}{lcccccc} y_0 & = & & & x_1 & & \\ y_1 & = & & & & x_2 & \\ y_2 & = & & & & & x_3 \\ y_3 & = & x_0 & + & x_1 & & \end{array}$$

or, in matrix terms, $y' = Ax'$, where A is the matrix

$$\begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 \end{pmatrix}.$$

(Here x' is the transpose of x, the column vector corresponding to the row vector x.)

The matrix A satisfies $A^{15} = I$, and no smaller power of A is equal to I. If V denotes the 4-dimensional vector space over the binary field, then for any non-zero vector $x \in V$, the fifteen vectors

$$x', Ax', A^2x', \ldots, A^{14}x'$$

are distinct and comprise all the non-zero vectors in V.

The connection between the polynomial and the matrix is simple:

> The polynomial of a shift register is equal to the characteristic polynomial (and to the minimal polynomial) of its matrix.

For, given a polynomial $f(x) = x^n + a_{n-1}x^{n-1} + \cdots + a_1x + a_0$, the *companion matrix* of f is defined to be the matrix

$$C(f) = \begin{pmatrix} 0 & 1 & 0 & \ldots & 0 & 0 \\ 0 & 0 & 1 & 0 & \ldots & 0 \\ \ddots & \ddots & \ddots & \ddots & \ddots & \ddots \\ 0 & 0 & 0 & \ldots & 0 & 1 \\ -a_0 & -a_1 & -a_2 & \ldots & -a_{n-2} & -a_{n-1} \end{pmatrix}$$

with zeros everywhere except for ones above the diagonal and the coefficients of f in reverse order with the sign changed in the bottom row. It is a standard result that the characteristic and minimal polynomials of $C(f)$ are both equal to f. Now over the binary field, $-a$ is the same as a, and the matrix associated with the shift register is precisely $C(f)$, so has the same characteristic and minimal polynomials.

We call a polynomial of degree n *primitive* if its associated shift register is primitive. Now the following theorem holds:

Theorem 7 *A primitive polynomial is irreducible.*

The proof of this theorem depends on the theory of finite fields and is beyond the scope of the course.

Example: Suppose that $n = 4$. How do we find all the primitive polynomials? First we find the irreducible polynomials. Let

$$f(x) = x^4 + ax_3 + bx^2 + cx + d$$

be a polynomial over $\mathbb{Z}/(2)$, so that all the coefficients are 0 or 1. There are $2^4 = 16$ polynomials altogether. Now, by the remainder theorem, if $f(0) = 0$, that is, $d = 0$, then x is a factor of $f(x)$; and if $f(1) = 0$, that is, $1 + a + b + c + d = 0$, then $x - 1$ is a factor (note that $x - 1$ is the same as $x + 1$). So we must have $d = 1$ and $a + b + c = 1$. Of the sixteen polynomials, just four pass these tests, namely

$$x^4 + x + 1, \quad x^4 + x^2 + 1, \quad x^4 + x^3 + 1, \quad x^4 + x^3 + x^2 + x + 1.$$

Now there is an irreducible polynomial of degree 2, namely $x^2 + x + 1$, and

$$(x^2 + x + 1)^2 = x^4 + x^2 + 1$$

is reducible. This leaves three polynomials. All of them are irreducible, since we have exhausted all the possible factorisations.

Now $x^4 + x + 1$ is primitive; this is the polynomial of the shift register with which we started. Similarly it can be checked that $x^4 + x^3 + 1$ is primitive. However, if we take the polynomial $x^4 + x^3 + x^2 + x + 1$ (with corresponding recurrence relation $x_{i+4} = x_{i+3} + x_{i+2} + x_{i+1} + x_i$), the starting configuration 0001 generates the sequence

$$000110001100011\ldots$$

of period 5. The other starting configurations also produce output of period 5. This polynomial is not primitive.

Using the theory of finite fields (which we sketch in the Appendix) it is possible to give formulae for the number of irreducible and primitive polynomials of degree n over $\mathbb{Z}/(2)$:

Theorem 8 *(a) The number of irreducible polynomials of degree n over $\mathbb{Z}/(2)$ is equal to*

$$\frac{1}{n}\sum_{d|n} 2^d \mu(n/d),$$

where μ is the Möbius function.

(b) The number of primitive polynomials of degree n over $\mathbb{Z}/(2)$ is equal to $\phi(2^n-1)/n$, where ϕ is Euler's function.

Euler's function was defined in Part 2 of the notes. The *Möbius function* μ is defined by

$$\mu(n) = \begin{cases} (-1)^k & \text{if } n \text{ is the product of } k \text{ distinct primes;} \\ 0 & \text{otherwise.} \end{cases}$$

For example, when $n = 4$, the number of irreducible polynomials is

$$\frac{1}{4}(2^4 - 2^2) = 3,$$

while the number of primitive polynomials is $\phi(15)/4 = 2$, in accordance with what we found above.

Golomb's Postulates

How do we tell if a sequence is random?

This is a very deep question, and several different solutions have been proposed. By definition, 'random' means 'selected from the set of all possible sequences, any sequence being equally likely', or (what amounts to the same thing, 'the symbols in the string are chosen independently with equal probability'. But this definition refers to the set of all possible sequences, and doesn't tell us anything about a single sequence. Indeed, any sequence can occur, even a constant sequence!

A completely different definition was proposed by Kolmogorov, who said:

> A sequence is random if it cannot be generated by an algorithm with a short description (i.e. much shorter than the sequence itself).

Using this definition, the keystring of the Fish cipher, or the string of digits of π, is not random. However, the definition is not easy to apply.

A more practical test was given by Golomb, who proposed three postulates. To state Golomb's postulates, we need a couple of definitions. Suppose that $a = a_0 a_1 \dots a_{n-1}$ is a binary sequence. We regard it as cyclic, so that a_0 is regarded as following a_{n-1}.

A *run* in the sequence is a subsequence such that all the entries are the same, which is as long as possible: that is, either a row of 1s with 0s at each end, or a row of 0s with 1s at each end. The *correlation* of two sequences a and b is defined to be $\sum a_i b_i$. The correlation of the sequence a with a cyclic shift of itself is called an *autocorrelation* of a; it is *in phase* if the shift is zero, and *out of phase* otherwise. Thus, the autocorrelation is $\sum a_i a_{i+m}$, where the subscripts are mod n; it is in phase if $m = 0$ and out of phase otherwise. (Sometimes in the literature a renormalisation is applied to the correlation; this doesn't affect the postulates below.)

Golomb's postulates are the following:

(G1) The numbers of 0s and 1s in the sequence are as near as possible to $n/2$ (that is, exactly $n/2$ if $n/2$ is even, and $(n \pm 1)/2$ if n is odd).

(G2) The number of runs of given length should halve when the length is increased by one (as long as possible), and where possible equally many runs of given length should consist of 0s as of 1s.

(G3) The out-of-phase autocorrelation should be constant (independent of the shift).

A sequence satisfying these postulates is called a *pseudo-noise sequence* or *PN-sequence*.

For example, consider the sequence

000100110101111

which we regard as being continued for ever in cyclic fashion. There are seven 1s and eight 0s, so (G1) is true. The runs are as follows:

- four of length 1, two 0s (beginning at positions 8 and 10) and two 1s (beginning at 3 and 9);
- two of length 2, one 00 (beginning at 4) and one 11 (beginning at 6);
- one of length 3, 000 beginning at 0;
- one of length 4, 1111 beginning at 11.

So (G2) is satisfied. For (G3), compare the sequence with each of its cyclic shifts:

000100110101111
100010011010111
110001001101011
111000100110101
111100010011010
011110001001101
101111000100110
010111100010011
101011110001001
110101111000100
011010111100010
001101011110001
100110101111000
010011010111100
001001101011110

We see by inspection that the autocorrelation of any two rows is equal to 4. Of course the in-phase autocorrelation is 8.

The sequences generated by shift registers are not of course random. In Kolmogorov's sense, they are very far from being random, since they are generated by a very simple machine. However, they are pseudo-noise sequences in Golomb's sense:

Theorem 9 *The output sequence of any primitive shift register satisfies Golomb's postulates.*

Proof The proof depends on the fact that, in one period of a primitive n-bit shift register, every non-zero n-tuple occurs exactly once, and the all-zero n-tuple never occurs.

To show postulate (G1): of the $2^n - 1$ possible non-zero n-tuples, $2^{n-1} - 1$ begin with zero and 2^{n-1} with one; so the cycle contains $2^{n-1} - 1$ zeros and 2^{n-1} ones, as required.

Postulate (G2) follows by a similar argument. To count runs of zeros and ones of length k, we have to count how many non-zero k-tupes begin with $01\ldots10$ or $10\ldots01$, with a run of k digits between two digits of the other kind. For $k < n-1$ there are obviously 2^{n-k-2} of each, since the remaining $n-k-2$ digits are arbitrary. The total number of runs of length k is thus 2^{n-k-1}, and this number halves each time we increase k by one (up to $n-2$).

We have to look at $k = n-1$ and $k = n$ separately. A run of n or more zeros cannot occur. So $n-1$ consecutive zeros must form a run, and this happens once (since the sequence $10\ldots0$ occurs once). A sequence of n ones occurs once, and there cannot be a longer sequence (since $n+1$ ones would contain two sequences of n), so n ones

form a run $01\ldots10$. It follows that there is no run of $n-1$ ones, since the first n terms $01\ldots1$ would agree with the first n terms in the run of length n.

Summarising, the numbers of runs are given in the table, where $k \leq n-2$:

k	runs of 0s	runs of 1s	total
k	2^{n-k-2}	2^{n-k-2}	2^{n-k-1}
$\ldots$	$\ldots$	$\ldots$	$\ldots$
$n-2$	1	1	2
$n-1$	1	0	1
n	0	1	1

So (G2) holds.

We will not prove (G3) here; the proof uses the theory of finite fields. In fact, the string of length 15 which we used in the preceding chapter is the output of the shift register with which we began this chapter.

Breaking a shift register

Although primitive shift registers have many good properties, such as satisfying Golomb's postulates, they have one fatal flaw: it doesn't take much information to break a stream cipher based on a shift register.

Theorem 10 *Suppose that a stream cipher is based on an n-bit shift register. Suppose that* $2n$ *consecutive bits of ciphertext and the corresponding plaintext are known. Then the cipher can be broken.*

Proof: From the $2n$ bits of ciphertext and corresponding plaintext, we obtain $2n$ consecutive bits of the keystring, say $u_0, u_1, \ldots, u_{2n-1}$. From Proposition 5, we have

$$\begin{aligned} u_n &= a_0u_0 + a_1u_1 + \cdots + a_{n-1}u_{n-1}, \\ u_{n+1} &= a_0u_1 + a_1u_2 + \cdots + a_{n-1}u_n, \\ &\cdots \\ u_{2n-1} &= a_0u_{n-1} + a_1u_n + \cdots + a_{n-1}u_{2n-2} \end{aligned}$$

This looks like a set of linear equations for the us, with the as as coefficients. But remember that in this case we know the us but not the as. So we regard them as equations for the unknowns $a_0, \ldots, a_{n-1}$. There are equally many equations as unknowns (namely n), and it is possible to show that the equations have a unique solution.

Thus we can determine the shift register, and then simulate its action (starting with the configuration $(u_0, \ldots, u_{n-1})$ to find the entire keystring.

The moral of the story is that any device that produces a long-period sequence from a small amount of data is vulnerable.

Example: Suppose that 11010110 is part of the output of a 4-bit shift register. We obtain the equations

$$\begin{array}{rcccccccc} 0 & = & a_0 & + & a_1 & & & + & a_3, \\ 1 & = & a_0 & & & + & a_2, & & \\ 1 & = & & & a_1 & & & + & a_3, \\ 0 & = & a_0 & & & + & a_2 & + & a_3. \end{array}$$

These equations have solution $a_0 = 1$, $a_1 = 0$, $a_2 = 0$, $a_3 = 1$. So the shift register has polynomial $x^4 + x^3 + 1$, and a period of its output is

1101011001000111

We see that the shift register is primitive.

How could $2n$ bits of plaintext be obtained? There are a number of methods. First of all, by guesswork. If Alice always starts her letters with "Dear Bob," we can make use of this fact. Another method would be to physically steal the plaintext from either Alice or Bob.

The breaking of the Fish cipher illustrates how Alice's carelessness can help Eve. The first step that led to the breaking of the Fish cipher occurred when the cryptanalysts discovered that two long messages had been enciphered using the same key (that is, the same settings and initial state of the wheels). Thus, we have

$$z = p \oplus k, \qquad z' = p' \oplus k,$$

where $\oplus$ here denotes bitwise binary addition. From the properties of binary addition, we deduce that

$$z \oplus z' = p \oplus p'.$$

This means that, when the two ciphertexts are added, the key disappears, and we have the sum of two plaintexts. Now these can be teased apart by frequency analysis, to find the two plaintexts p and p'. Now we can find the key $k = p \oplus z$. The cryptanalysts used the key to deduce the structure of the cipher machine. This is similar to (but rather more complicated than) our use of $2n$ bits of key to break an n-bit shift register.

Worked example The *seven-bit ASCII code* represents letters, digits, and punctuation as characters from the set of integers in the range $32 \ldots 127$; the capital letters `A...Z` are represented by $65 \ldots 90$, and lower-case letters `a...z` by $97 \ldots 112$. Integers in the range $0 \ldots 31$ are used for control codes. The integers are then written in base 2, as 7-tuples of zeros and ones.

You intercept the string

000011011011101010111111011101001101111001001111000010110001010101010101

You have reason to believe that it is a message in seven-bit ASCII encrypted by means of a stream cipher based on a seven-bit shift register, and that the first two letters of the message are Su. Decrypt the string.

Solution The 7-bit ASCII code for Su is 10100111110101. Subtracting these fourteen bits of plaintext from the first fourteen bits of ciphertext gives us fourteen bits of key: 10101010011011. So the equations for the shift register are

$$\begin{array}{rcccccccccccc}
0 & = & a_0 & & + & a_2 & & + & a_4 & & & + & a_6 \\
0 & = & & a_1 & & & + & a_3 & & + & a_5 & & \\
1 & = & a_0 & & + & a_2 & & + & a_4 & & & & \\
1 & = & & a_1 & & & + & a_3 & & & & + & a_6 \\
0 & = & a_0 & & + & a_2 & & & & + & a_5 & + & a_6 \\
1 & = & & a_1 & & & & + & a_4 & + & a_5 & & \\
1 & = & a_0 & & & & + & a_3 & + & a_4 & & + & a_6
\end{array}$$

Solving, we find $(a_0,\ldots,a_6) = (1,1,0,1,0,0,1)$, so the shift register polynomial is $x^7+x^6+x^3+x+1$. Now we can continue the key to 70 bits using the recurrence relation $x_{n+7} = x_{n+6}+x_{n+3}+x_{n+1}+x_n$ and subtract it from the ciphertext to obtain the plaintext, and then divide the plaintext into 7-bit blocks and decode each block to obtain the message: `Surrender!`

Appendix: Finite fields

This material is not part of the course. But any serious investigation of shift registers must observe that they are very closely connected with finite fields. A field is a set with two operations (addition and multiplication) in which the 'usual rules' apply. For example, the rational, real or complex numbers, or the integers modulo p (where p is prime) are fields.

The finite fields were classified by Galois around 1830:

Theorem 11 *The order of a finite field must be a prime power. For every prime power q, there is a field with q elements, and it is unique up to isomorphism.*

The field with q elements is denoted by $\mathrm{GF}(q)$ (for 'Galois field') in honour of Galois.

Two properties of finite fields are important here:

Theorem 12 *The multiplicative group of a finite field is cyclic.*

This means that $\mathrm{GF}(q)$ contains an element α with the property that all the $q-1$ non-zero elements are powers of α. Thus, $\alpha^{q-1}=1$, but no smaller power of α is equal to 1. Such an element α is said to be a *primitive element* of $\mathrm{GF}(q)$. The number of primitive elements of $\mathrm{GF}(q)$ is equal to $\phi(q-1)$, where ϕ is Euler's function.

Theorem 13 *Let p and p_1 be primes. The field $\mathrm{GF}(p^n)$ contains a subfield $\mathrm{GF}(p_1^m)$ if and only if $p=p_1$ and m divides n. In this case, there is a unique subfield $\mathrm{GF}(p^m)$ of $\mathrm{GF}(p^n)$.*

Now let q be a given prime power. The field $\mathrm{GF}(q^n)$ contains a unique subfield $\mathrm{GF}(q)$. For each element $\theta \in \mathrm{GF}(q^n)$, there is a *minimal polynomial* of θ over $\mathrm{GF}(q)$, that is, a monic polynomial satisfied by θ. This polynomial is always irreducible, and its degree is equal to m if the smallest subfield of $\mathrm{GF}(q^n)$ containing $\mathrm{GF}(q)$ and θ is $\mathrm{GF}(q^m)$.

The monic polynomial of θ has degree n if and only if θ lies in no subfield of $\mathrm{GF}(q^n)$ containing $\mathrm{GF}(q)$ (except $\mathrm{GF}(q^n)$ itself). Every irreducible polynomial of degree n over $\mathrm{GF}(q)$ is the minimal polynomial of exactly n elements of $\mathrm{GF}(q^n)$.

Now consider the case where $q=2$. We begin by reversing the procedure and constructing $\mathrm{GF}(2^4)$ as an example. Let α be a root of the irreducible polynomial x^4+x+1 over $\mathrm{GF}(2)$. Thus, $\alpha^4+\alpha+1=0$, or (since $-1=+1$) $\alpha^4=\alpha+1$. We can make a table of powers of α as follows:

$$
\begin{array}{lcccccccc}
\alpha^0 & = & & & & & & & 1 \\
\alpha^1 & = & & & & & \alpha & & \\
\alpha^2 & = & & & \alpha^2 & & & & \\
\alpha^3 & = & \alpha^3 & & & & & & \\
\alpha^4 & = & & & & & \alpha & + & 1 \\
\alpha^5 & = & & & \alpha^2 & + & \alpha & & \\
\alpha^6 & = & \alpha^3 & + & \alpha^2 & & & & \\
\alpha^7 & = & \alpha^3 & & & + & \alpha & + & 1 \\
\alpha^8 & = & & & \alpha^2 & & & + & 1 \\
\alpha^9 & = & \alpha^3 & & & +\alpha & & & \\
\alpha^{10} & = & & & \alpha^2 & + & \alpha & + & 1 \\
\alpha^{11} & = & \alpha^3 & + & \alpha^2 & + & \alpha & & \\
\alpha^{12} & = & \alpha^3 & + & \alpha^2 & + & \alpha & + & 1 \\
\alpha^{13} & = & \alpha^3 & + & \alpha^2 & & & + & 1 \\
\alpha^{14} & = & \alpha^3 & & & & & + & 1
\end{array}
$$

and $\alpha^{15} = 1 = \alpha^0$, so the sequence repeats (like the shift register). We see that α is a primitive element of the field $\mathrm{GF}(2^4)$; the field consists of zero and the fifteen powers of α.

Using this table as a table of logarithms, we can do arithmetic in the field. For example,

$$\begin{aligned}(\alpha^2+\alpha+1)+(\alpha^3+\alpha^2+\alpha) &= \alpha^3+1,\\ (\alpha^2+\alpha+1)\cdot(\alpha^3+\alpha^2+\alpha) &= \alpha^{10}\cdot\alpha 11 = \alpha^6 = \alpha^3+\alpha^2.\end{aligned}$$

Now let $\beta = \alpha^7$. We have

$$\begin{aligned}\beta^2 &= \alpha^{14} = \alpha^3+1,\\ \beta^3 &= \alpha^6 = \alpha^3+\alpha^2,\\ \beta^4 &= \alpha^{13} = \alpha^3+\alpha^2+1.\end{aligned}$$

So we see that $\beta^4 = \beta^3+1$, so that β satisfies the primitive polynomial x^4+x^3+1.

Similarly we find that $\gamma = \alpha^3$ satisfies the irreducible but not primitive polynomial $x^4+x^3+x^2+x+1$, while $\delta = \alpha^5$ has minimal polynomial x^2+x+1 and lies in a subfield $\mathrm{GF}(4)$ consisting of the elements $0, 1, \alpha^5, \alpha^{10}$.

The three irreducible polynomials of degree 4 each have four roots. The irreducible polynomial x^2+x+1 has two roots. The two elements $0, 1$ have minimal polynomials x and $x+1$ respectively of degree 1. Thus all elements of $\mathrm{GF}(16)$ are accounted for.

Theorem 14 *Let θ be an element of $\mathrm{GF}(2^n)$ with minimal polynomial $f(x)$ of degree n. Then $f(x)$ is a primitive polynomial (in the sense that the associated shift register has period 2^n-1) if and only if θ is a primitive element of $\mathrm{GF}(2^n)$.*

For example, suppose that $n = 4$. The proper subfields of $\mathrm{GF}(16)$ are

$$\mathrm{GF}(2) \subseteq \mathrm{GF}(4) \subseteq \mathrm{GF}(16),$$

where $\mathrm{GF}(2)$ is the binary field $\mathbb{Z}/(2)$. So there are 12 elements of $\mathrm{GF}(16)$ which lie in no proper subfield, and thus $12/4 = 3$ irreducible polynomials of degree 4. Moreover, there are $\phi(15) = 2\cdot 4 = 8$ primitive elements of $\mathrm{GF}(16)$, and hence $8/4 = 2$ primitive polynomials. These agree with what we found by hand earlier.

The formulae for the number of irreducible and primitive polynomials given in Theorem 8 follow from these considerations. (We need to use the "inclusion-exclusion principle" from Combinatorics to count the irreducible polynomials, since we have to count elements in $\mathrm{GF}(2^n)$ which do not lie in any proper subfield. The argument for primitive polynomials is more direct.)